Algorithmique et structures de données I

Riadh Ben Messaoud

Université 7 novembre à Carthage

Faculté des Sciences Économiques et de Gestion de Nabeul

$1^{\text{ère}}$ année Licence Fondamentale IAG

$1^{\text{ère}}$ année Licence Appliquée IAG

Année universitaire
2009 – 2010

Syllabus du cours ...

Syllabus : http://eric.univ-lyon2.fr/~rbenmessaoud/

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : Connaissances générales en informatique utiles, mais pas indispensables.

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Syllabus du cours ...

Syllabus : `http://eric.univ-lyon2.fr/~rbenmessaoud/`

Objectif :

- se familiariser avec les méthodes de résolution de problèmes avec l'outil informatique ;
- apprendre les principes de l'algorithmique ;
- acquérir un début de maîtrise des techniques et langages de programmation.

Pré-requis : **Connaissances générales en informatique utiles, mais pas indispensables.**

Organisation : 21 h de cours + 21 h de TD

Bibliographie :

- *Introduction à l'algorithmique, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, Dunod, Paris, 2004.*
- *Algorithmique Application en C, Jean-Michel Léry, Pearson Education, 2005.*
- *Algorithmique et programmation en Java, Vincent Granet, Dunod, Paris, 2000.*
- *Débuter en programmation, Greg Perry, CampusPress, 2003.*

Plan du cours

1. Introduction
2. Environnement algorithmique
3. Variables
4. Structures conditionnelles
5. Structures itératives
6. Tableaux
7. Sous-programmes
8. Mode de passage de paramètres

Plan du cours

1. Introduction
2. Environnement algorithmique
3. Variables
4. Structures conditionnelles
5. Structures itératives
6. Tableaux
7. Sous-programmes
8. Mode de passage de paramètres

Qu'est ce qu'un algorithme ?

Définition informelle

Un algorithme est une procédure de **calcul bien définie** qui prend en **entrée** une valeur, ou un ensemble de valeurs, et qui donne en **sortie** une valeur, ou un ensemble de valeurs. Un algorithme est donc une **séquence d'étapes** de calcul qui **transforment** l'entrée en sortie.

Une autre définition

Un algorithme est un moyen pour un humain de présenter la résolution par **calcul** d'un problème à une autre personne physique (un autre humain) ou virtuelle (un calculateur). En effet, un algorithme est un énoncé dans un **langage** bien défini d'une **suite d'opérations** permettant de **résoudre** par calcul un problème.

Encore une autre définition, plus générale

Un algorithme est une **suite d'instructions**, qui une fois **exécutée** correctement, conduit à un **résultat donné**.

Qu'est ce qu'un algorithme ?

Définition informelle

Un algorithme est une procédure de **calcul bien définie** qui prend en **entrée** une valeur, ou un ensemble de valeurs, et qui donne en **sortie** une valeur, ou un ensemble de valeurs. Un algorithme est donc une **séquence d'étapes** de calcul qui **transforment** l'entrée en sortie.

Une autre définition

Un algorithme est un moyen pour un humain de présenter la résolution par **calcul** d'un problème à une autre personne physique (un autre humain) ou virtuelle (un calculateur). En effet, un algorithme est un énoncé dans un **langage** bien défini d'une **suite d'opérations** permettant de **résoudre** par calcul un problème.

Encore une autre définition, plus générale

Un algorithme est une **suite d'instructions**, qui une fois **exécutée** correctement, conduit à un **résultat donné**.

Qu'est ce qu'un algorithme ?

Définition informelle

Un algorithme est une procédure de **calcul bien définie** qui prend en **entrée** une valeur, ou un ensemble de valeurs, et qui donne en **sortie** une valeur, ou un ensemble de valeurs. Un algorithme est donc une **séquence d'étapes** de calcul qui **transforment** l'entrée en sortie.

Une autre définition

Un algorithme est un moyen pour un humain de présenter la résolution par **calcul** d'un problème à une autre personne physique (un autre humain) ou virtuelle (un calculateur). En effet, un algorithme est un énoncé dans un **langage** bien défini d'une **suite d'opérations** permettant de **résoudre** par calcul un problème.

Encore une autre définition, plus générale

Un algorithme est une **suite d'instructions**, qui une fois **exécutée** correctement, conduit à un **résultat donné**.

Qu'est ce qu'un algorithme ?

Définition informelle

Un algorithme est une procédure de **calcul bien définie** qui prend en **entrée** une valeur, ou un ensemble de valeurs, et qui donne en **sortie** une valeur, ou un ensemble de valeurs. Un algorithme est donc une **séquence d'étapes** de calcul qui **transforment** l'entrée en sortie.

Une autre définition

Un algorithme est un moyen pour un humain de présenter la résolution par **calcul** d'un problème à une autre personne physique (un autre humain) ou virtuelle (un calculateur). En effet, un algorithme est un énoncé dans un **langage** bien défini d'une **suite d'opérations** permettant de **résoudre** par calcul un problème.

Encore une autre définition, plus générale

Un algorithme est une **suite d'instructions**, qui une fois **exécutée** correctement, conduit à un **résultat donné**.

Remarque

On désigne par **algorithmique** ou **algorithmie** l'ensemble des activités logiques qui relèvent des **algorithmes**.

Exemples d'algorithmes :

- indiquer un chemin à un touriste égaré ;
- rédiger une recette de cuisine ;
- élaborer un mode d'emploi pour faire fonctionner un magnétoscope ;
- etc.

Important

Pour fonctionner, un algorithme doit contenir uniquement des instructions **compréhensibles** par celui qui devra l'exécuter.

Remarque

On désigne par **algorithmique** ou **algorithmie** l'ensemble des activités logiques qui relèvent des **algorithmes**.

Exemples d'algorithmes :

- indiquer un chemin à un touriste égaré ;
- rédiger une recette de cuisine ;
- élaborer un mode d'emploi pour faire fonctionner un magnétoscope ;
- etc.

Important

Pour fonctionner, un algorithme doit contenir uniquement des instructions **compréhensibles** par celui qui devra l'exécuter.

Remarque

On désigne par **algorithmique** ou **algorithmie** l'ensemble des activités logiques qui relèvent des **algorithmes**.

Exemples d'algorithmes :

- indiquer un chemin à un touriste égaré ;
- rédiger une recette de cuisine ;
- élaborer un mode d'emploi pour faire fonctionner un magnétoscope ;
- etc.

Important

Pour fonctionner, un algorithme doit contenir uniquement des instructions **compréhensibles** par celui qui devra l'exécuter.

- **XVIII^{ème} siècle av. J.-C.** : les **Babyloniens** ont défini des descriptions exhaustives d'algorithmes pour des calculs concernant le commerce et les impôts ;

- **XVIII^{ème} siècle av. J.-C.** : les **Babyloniens** ont défini des descriptions exhaustives d'algorithmes pour des calculs concernant le commerce et les impôts ;

- **III^{ème} siècle av. J.-C.** : **Euclide** a introduit (dans son ouvrage les *Éléments*) le célèbre algorithme qui permet de trouver le plus grand diviseur commun (PGCD) de deux nombres ;

- **IX^{ème} siècle** : **Al Khuwarizmi** été le premier qui a formalisé la notion d'algorithme dans son ouvrage *L'algèbre et le balancement* ;

- **XII^{ème} siècle** : **Adelard de Bath** a introduit le terme latin de **algorismus** (par référence au nom de **Al Khuwarizmi**). Ce mot donne **algorithme** en français en 1554.

Faut-il être matheux pour être bon en algorithmique ?

Faut-il être matheux pour être bon en algorithmique ?

Non, pas du tout !

Faut-il être matheux pour être bon en algorithmique ?

Non, pas du tout !

La maîtrise de l'algorithmique requiert trois qualités :

1. Il faut être **méthodique**. Avant d'écrire les instructions d'un algorithme, il faut analyser le **problème** à résoudre. Il faut ensuite définir les **entrées** et les **sorties** de l'algorithme.

2. Il faut avoir de l'**intuition**. Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. Les réflexes du raisonnement algorithmique deviennent **spontanés** avec l'**expérience**.

3. Il faut être **rigoureux**. Chaque fois qu'on écrit une série d'instructions, il faut **systématiquement** se mettre mentalement à la place de la machine qui va les exécuter. Si nécessaire, il faut avoir recours à une simulation sur papier.

Faut-il être matheux pour être bon en algorithmique ?

Non, pas du tout !

La maîtrise de l'algorithmique requiert trois qualités :

1. Il faut être **méthodique**. Avant d'écrire les instructions d'un algorithme, il faut analyser le **problème** à résoudre. Il faut ensuite définir les **entrées** et les **sorties** de l'algorithme.

2. Il faut avoir de l'**intuition**. Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. Les réflexes du raisonnement algorithmique deviennent **spontanés** avec l'**expérience**.

3. Il faut être **rigoureux**. Chaque fois qu'on écrit une série d'instructions, il faut **systématiquement** se mettre mentalement à la place de la machine qui va les exécuter. Si nécessaire, il faut avoir recours à une simulation sur papier.

Faut-il être matheux pour être bon en algorithmique ?

Non, pas du tout !

La maîtrise de l'algorithmique requiert trois qualités :

1. Il faut être **méthodique**. Avant d'écrire les instructions d'un algorithme, il faut analyser le **problème** à résoudre. Il faut ensuite définir les **entrées** et les **sorties** de l'algorithme.

2. Il faut avoir de l'**intuition**. Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. Les réflexes du raisonnement algorithmique deviennent **spontanés** avec l'**expérience**.

3. Il faut être **rigoureux**. Chaque fois qu'on écrit une série d'instructions, il faut **systématiquement** se mettre mentalement à la place de la machine qui va les exécuter. Si nécessaire, il faut avoir recours à une simulation sur papier.

Faut-il être matheux pour être bon en algorithmique ?

Non, pas du tout !

La maîtrise de l'algorithmique requiert trois qualités :

1. Il faut être **méthodique**. Avant d'écrire les instructions d'un algorithme, il faut analyser le **problème** à résoudre. Il faut ensuite définir les **entrées** et les **sorties** de l'algorithme.

2. Il faut avoir de l'**intuition**. Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. Les réflexes du raisonnement algorithmique deviennent **spontanés** avec l'**expérience**.

3. Il faut être **rigoureux**. Chaque fois qu'on écrit une série d'instructions, il faut **systématiquement** se mettre mentalement à la place de la machine qui va les exécuter. Si nécessaire, il faut avoir recours à une simulation sur papier.

Les ordinateurs ne comprennent que **quatre catégories** d'instructions :

Les ordinateurs ne comprennent que **quatre catégories** d'instructions :

1. **l'affectation de variables** ;
2. **la lecture/écriture** ;
3. **les tests** (les structures conditionnelles) ;
4. **les boucles** (les structures itératives).

Les ordinateurs ne comprennent que **quatre catégories** d'instructions :

1. **l'affectation de variables** ;
2. **la lecture/écriture** ;
3. **les tests** (les structures conditionnelles) ;
4. **les boucles** (les structures itératives).

Important

Un algorithme informatique se ramène toujours à la combinaison de ces quatre types d'instruction. Il peut y en avoir quelques unes, quelques dizaines, et jusqu'à plusieurs centaines de milliers.

Plan du cours

1. Introduction
2. **Environnement algorithmique**
3. Variables
4. Structures conditionnelles
5. Structures itératives
6. Tableaux
7. Sous-programmes
8. Mode de passage de paramètres